(Ab)using Linux as a Trusted Bootloader

Eric Richter
erichte@linux.vnet.ibm.com

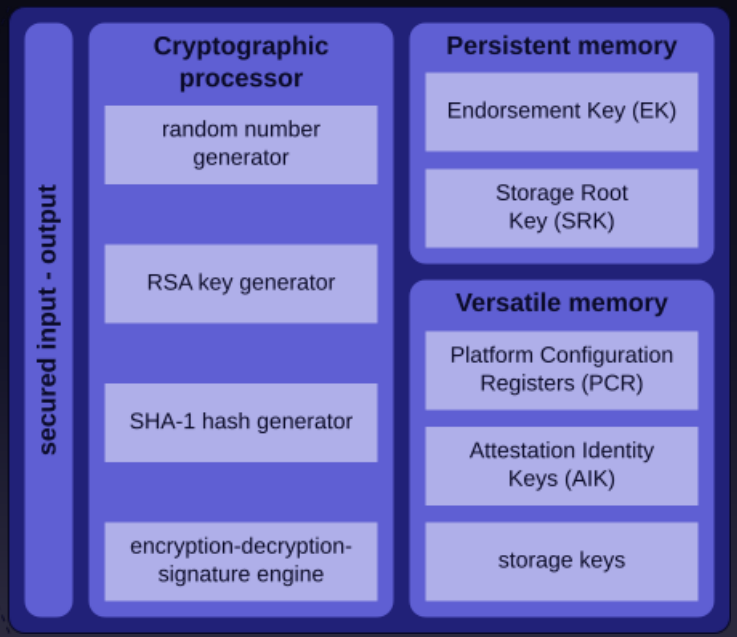
Overview

1. Background Info
   - TPM
   - Trusted Boot
2. Petitboot
3. “Trusted” Petitboot
4. Kernel Changes
5. Questions

Background Information

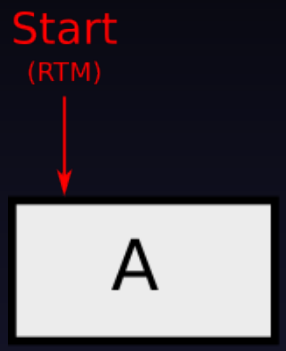
Trusted Platform Module (TPM)

- Small, inexpensive security cryptoprocessor
- Contains a bank of “append only” data registers called PCRs

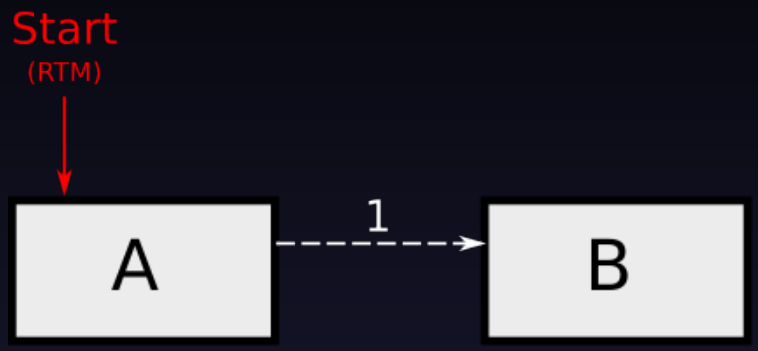


TPM 1.2 Figure by Guillaume Piolle

Trusted Boot

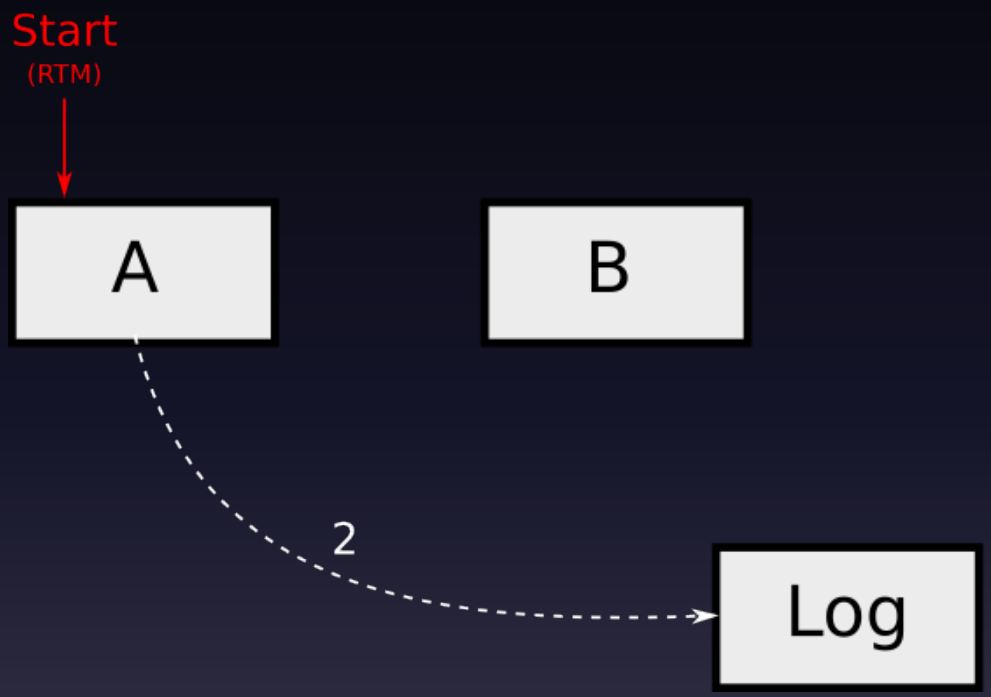


Trusted Boot



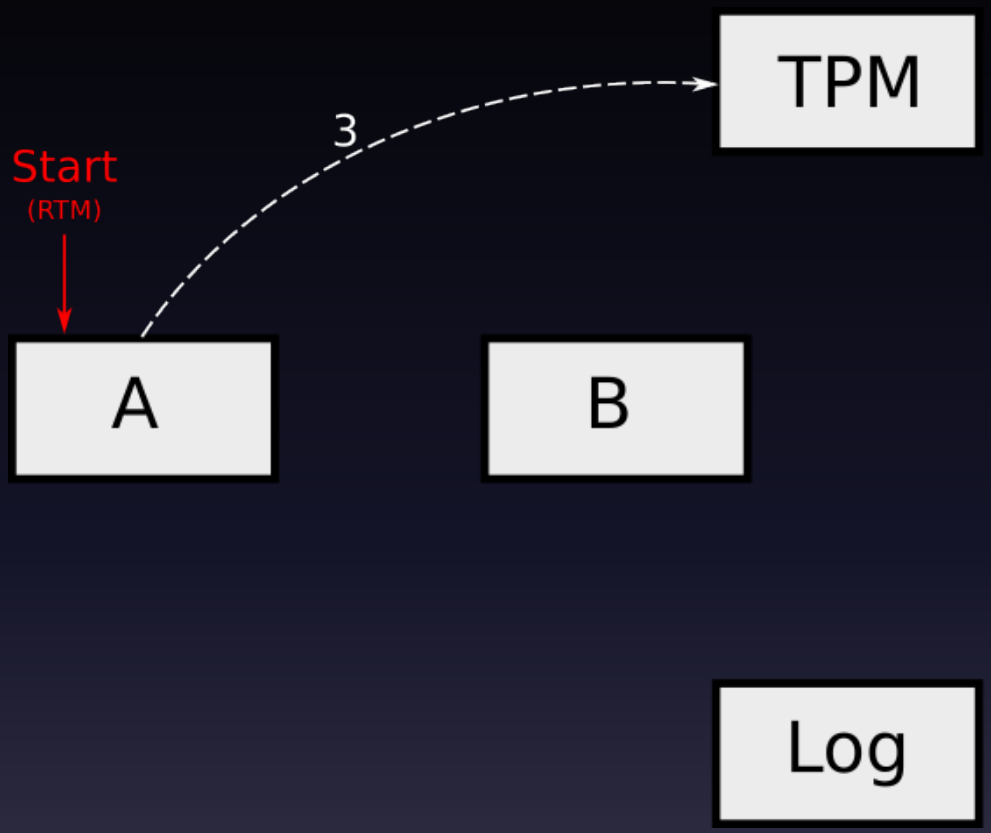
1. Measure next component

Trusted Boot



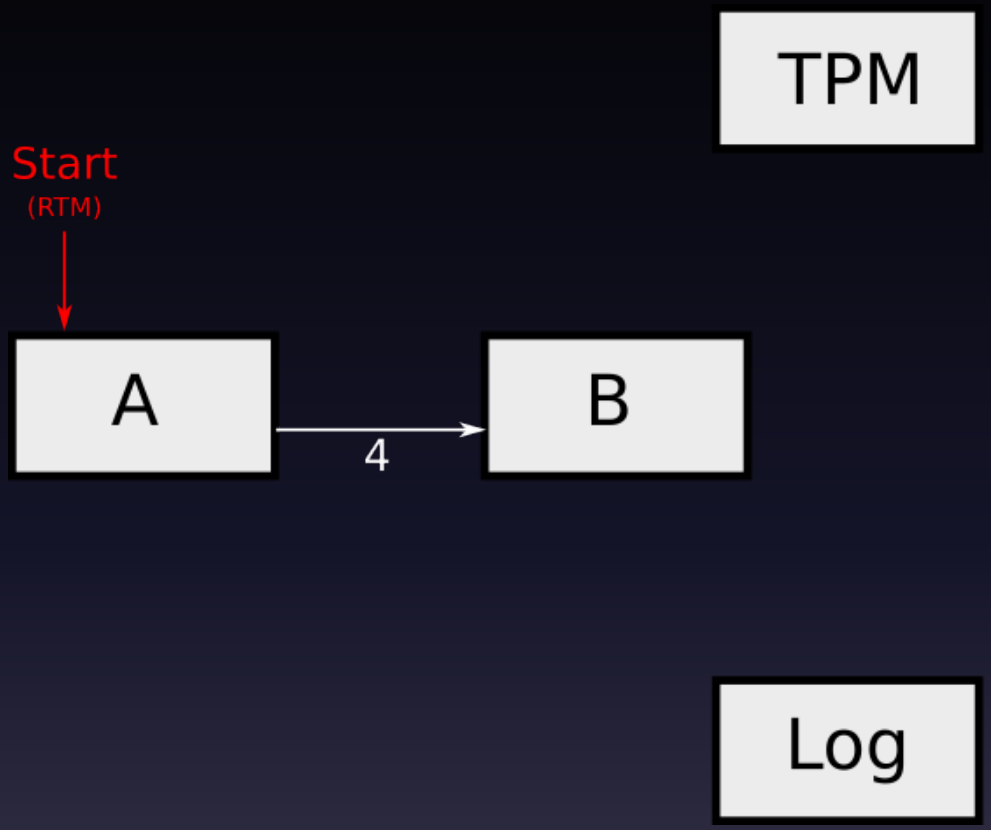
1. Measure next component
2. Log the Event

Trusted Boot



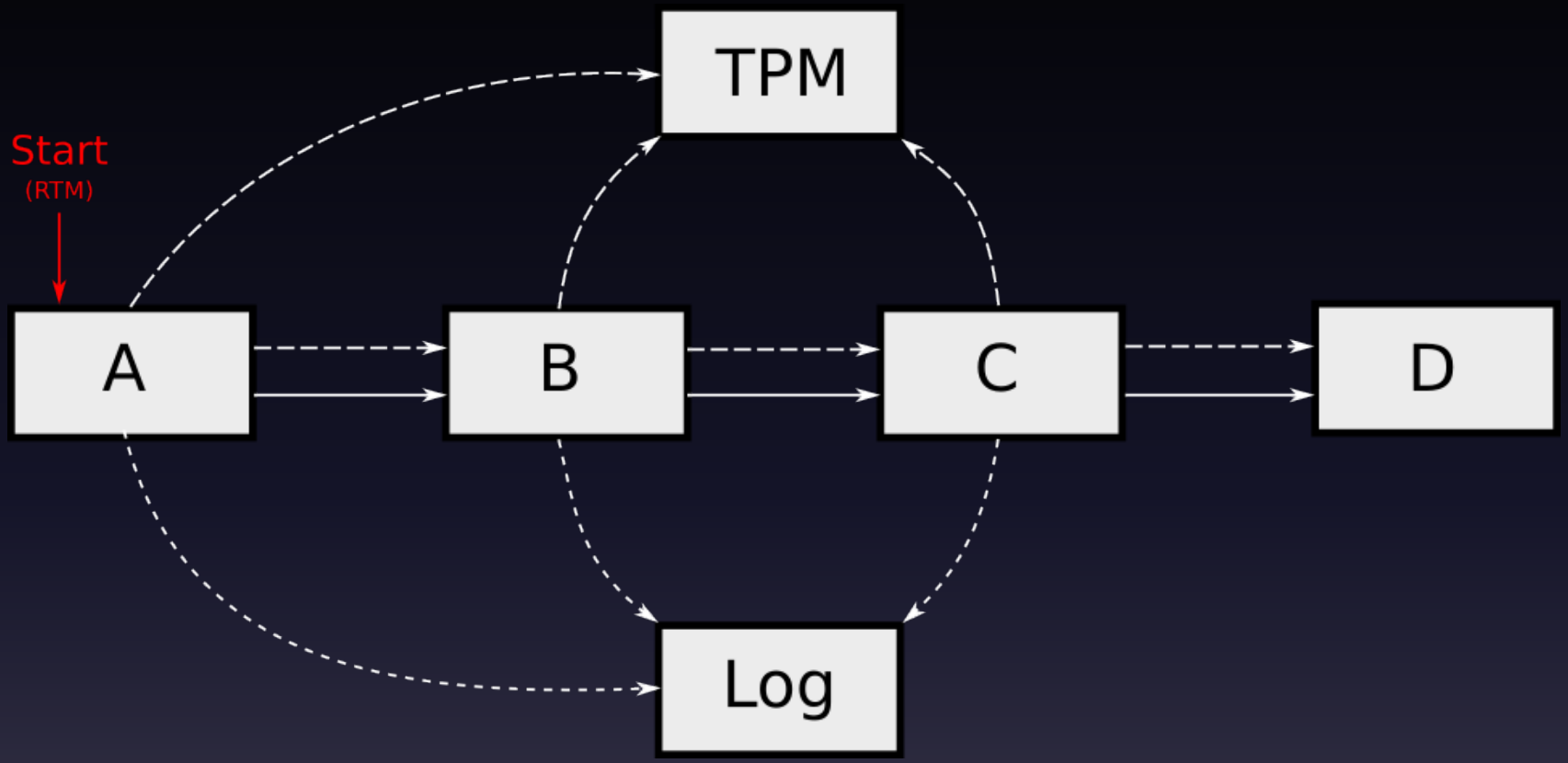
1. Measure next component
2. Log the Event
3. Extend PCR

Trusted Boot



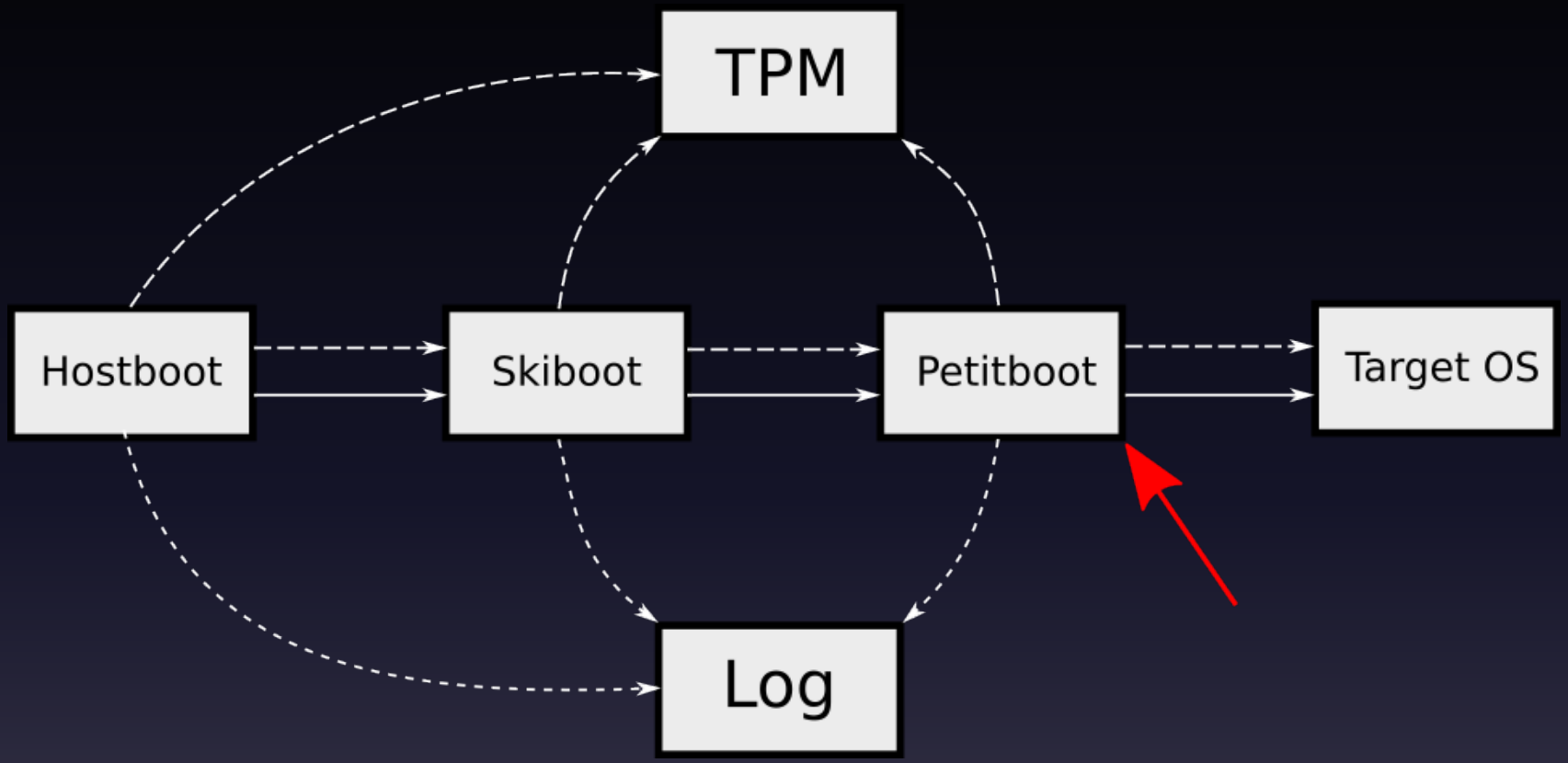
1. Measure next component
2. Log the Event
3. Extend PCR
4. Transfer Execution

Trusted Boot



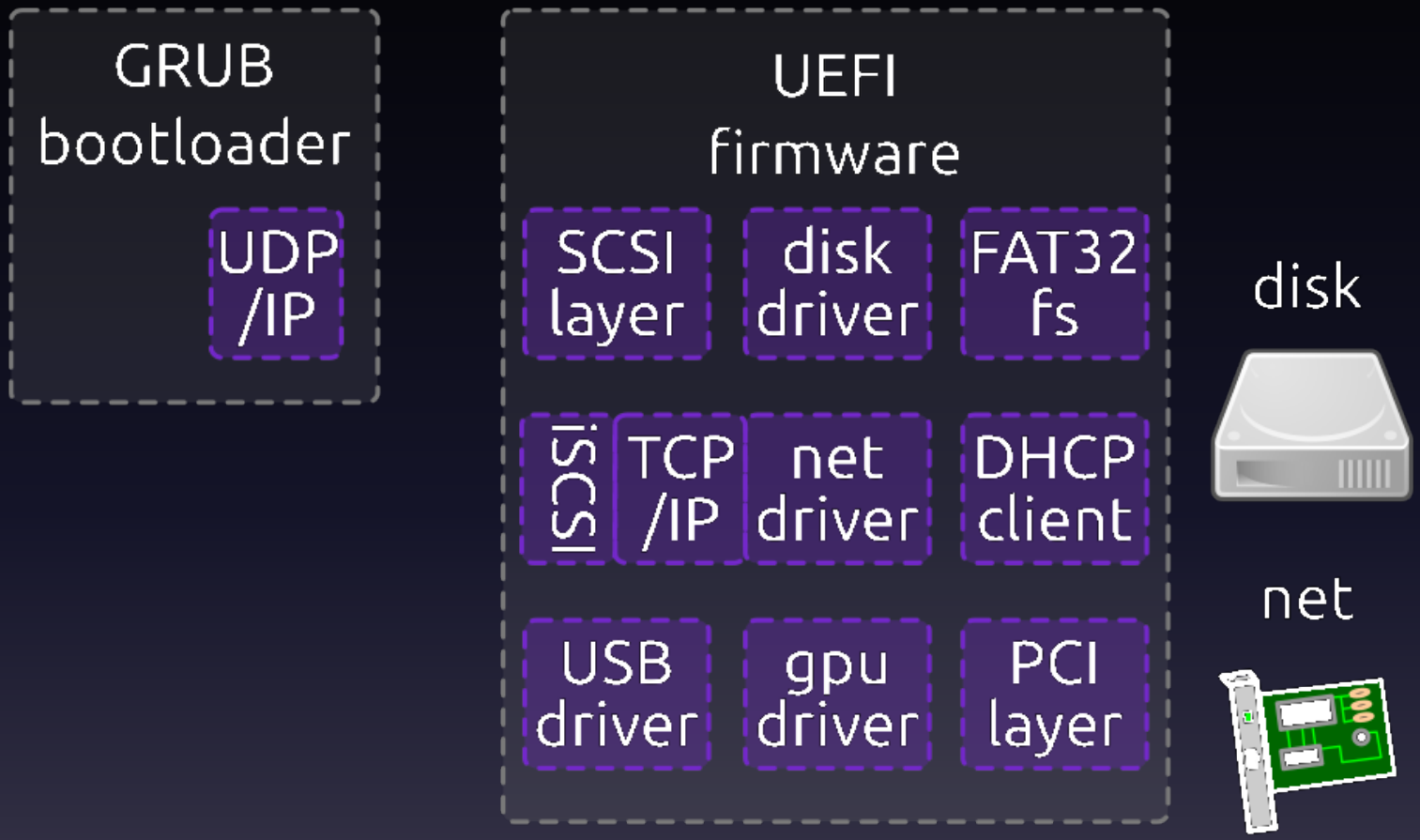
1. Measure next component
2. Log the Event
3. Extend PCR
4. Transfer Execution
5. Repeat

Trusted Boot on OpenPOWER

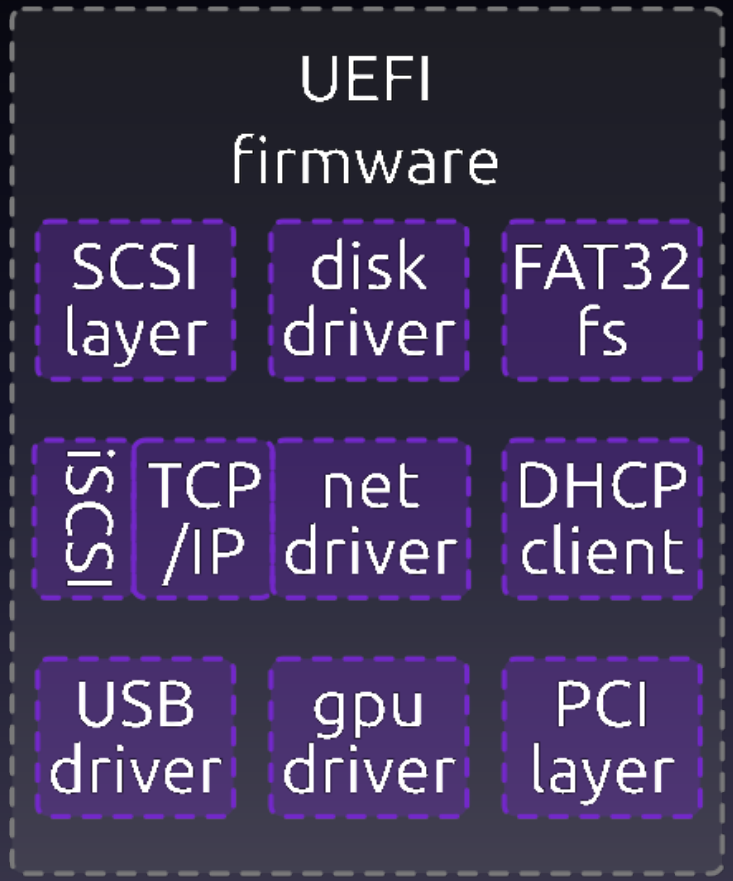


What is Petitboot?

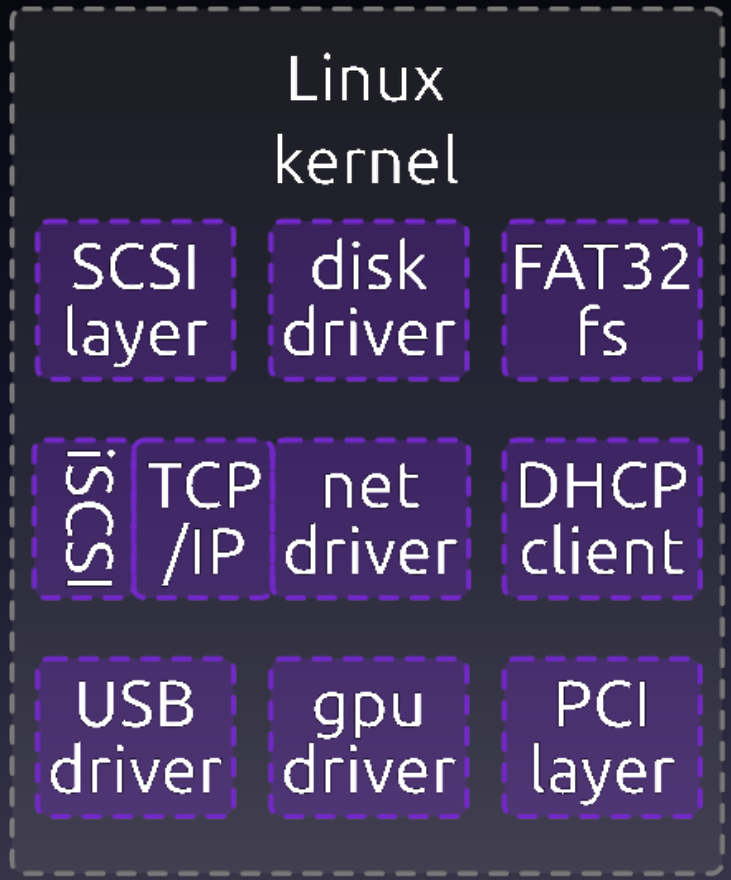
UEFI + GRUB


GRUB
bootloader
UDP
/IP
UEFI
firmware
SCSI
layer
disk
driver
FAT32
fs
iSCSI
TCP
/IP
net
driver
DHCP
client
USB
driver
gpu
driver
PCI
layer
disk
net

The Wheel


UEFI
firmware
SCSI layer
disk driver
FAT32 fs
iSCSI
TCP /IP
net driver
DHCP client
USB driver
gpu driver
PCI layer

Reinvented?

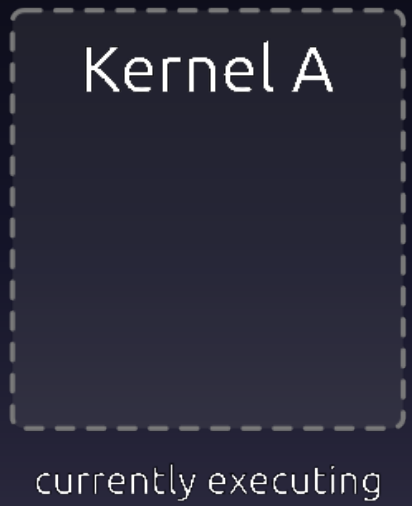

Linux
kernel
SCSI
layer
disk
driver
FAT32
fs
iSCSI
TCP
/IP
net
driver
DHCP
client
USB
driver
gpu
driver
PCI
layer

Petitboot

Linux Kernel based bootloader via kexec

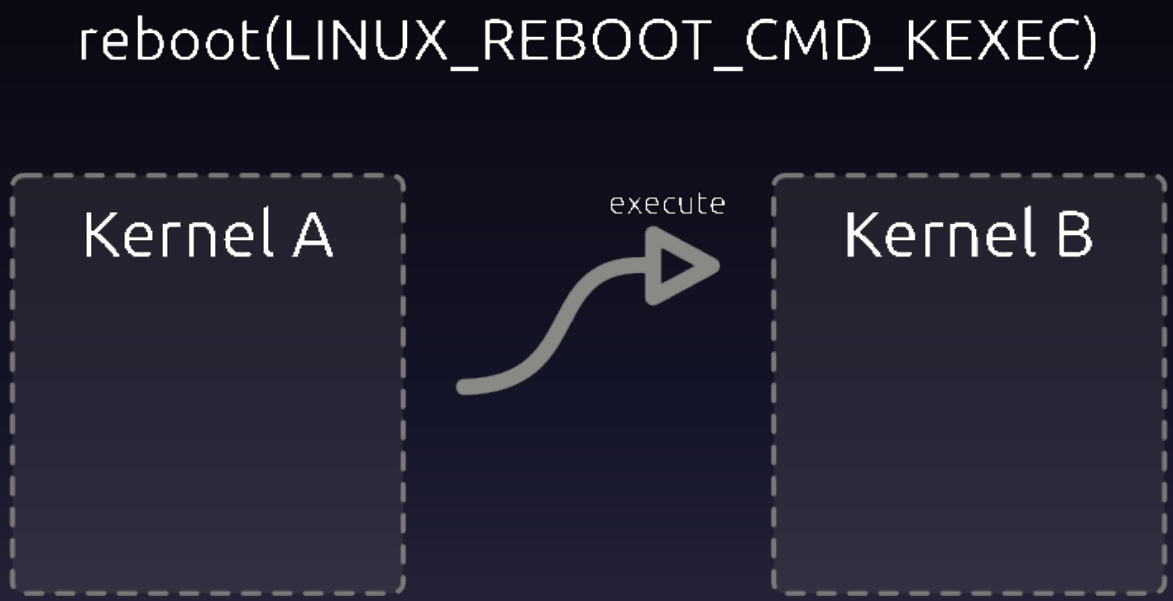
- Reuse kernel drivers
- Provides userspace, simplifying development
- Platform independence
- Well tested ;)

Kexec



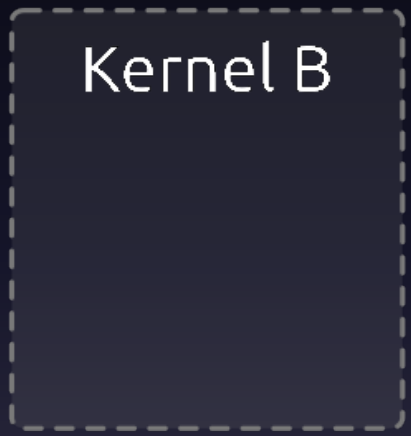
Kexec

Kexec



Kexec

Kexec



“Trusted” Petitboot?

Measure in Petitboot!

Petitboot is a userspace application, take the measurements in there!

Except...

- Need some form of TCG Software Stack (TSS)
- Need write access to the event log
- Exit to shell $\rightarrow$ manual kexec

Measure in kexec-{tools,lite}...

Okay, so modify the kexec binary to take the measurement.

Well, that fixes the manual kexec issue, but...

- Still needs some kind of TSS
- Still need to append the log

"Trusted" Petitboot?

Trusted measurements provide a new set of challenges in a kernel environment.

Would need...

- ...a TPM driver
- ...a way to interface with the TPM
- ...a way to measure the next component
- ...a way to log the measurement
- ...for this to be upstreamable

If only we had an integrity subsystem...

"Trusted" Petitboot.

Oh wait, there's IMA.

IMA can handle...

- ...talking to the TPM via the device driver
- ...extending the data
- ...measuring the next component
- ...logging what was measured
- ...some small modifications

Integrity Measurement Architecture

Measures and appraises files based on set policy rules.
Example rule to measure files opened by root:

```
measure func=FILE_CHECK mask=MAY_READ uid=0
```

Logs them like so:

```
10 1d8d532d463c9f8c205d0df7787669a85f93e260 ima-ng sha1:0000000000000000000000000000000000000000 boot_aggregate
10 ef411bae164fd624ea94fc9ef82f892c82d78dcd ima-ng sha1:bbe98e20b850f3907611fb96354b5e007a9179f4 /init
10 5a793eedd4512ea7cb509a6ccae4d3d804eb648a ima-ng sha1:cbad4f2990812051037d0b173878b71dfa53fba5 /bin/busybox
10 52c1530e65dde06cc50b5cfab6fc8517536dc4e5 ima-ng sha1:8fb2fcb2e0cc7523f15c9712b203e97793703cd1 /lib64/ld64.so.2
10 97a68f06ce0c81aa32e9be05432fd6a0245d88d4 ima-ng sha1:1e763bf47b638f72c289c019aa5e92c721a1285a /etc/ld.so.cache
```

Kernel Changes

Well, it's not a perfect solution.

Still need...

1. IMA Hook for measuring kernel, initramfs
2. Support for extending alternate PCRs
3. Preservation of the IMA log across kexec
4. Device tree passing and measurement?

1. Kernel & Initramfs Hooks

Problem: Need to measure kernel and initramfs

Solution: As of 4.6, these are already included!

Example policy:

```
measure func=KEXEC_KERNEL_CHECK
measure func=KEXEC_INITRAMFS_CHECK
```

2. Extending Other PCRs

```
10 1d8d532d463c9f8c205d0df7787669a85f93e260 ima-ng sha1:0000000000000000000000000000000000000000 boot_aggregate
10 ef411bae164fd624ea94fc9ef82f892c82d78dcd ima-ng sha1:bbe98e20b850f3907611fb96354b5e007a9179f4 /init
10 5a793eedd4512ea7cb509a6ccae4d3d804eb648a ima-ng sha1:cbad4f2990812051037d0b173878b71dfa53fba5 /bin/busybox
10 52c1530e65dde06cc50b5cfab6fc8517536dc4e5 ima-ng sha1:8fb2fcb2e0cc7523f15c9712b203e97793703cd1 /lib64/ld64.so.2
10 97a68f06ce0c81aa32e9be05432fd6a0245d88d4 ima-ng sha1:1e763bf47b638f72c289c019aa5e92c721a1285a /etc/ld.so.cache
```

↑

Problem: IMA measures into a PCR as defined by Kconfig.

Solution: Add flexibility into the policy

```
measure func=FILE_CHECK mask=MAY_READ uid=0 pcr=11
                                            ↑
```

```
11 ce587f7a3f6a4df6223c471e717e18c8d8d573a0 ima-ng sha1:e3226a47630f87f23b358b298c35c26d40a61379 /etc/sudoers
11 439fce9e817a2867584665644f01eaed87d4fa03 ima-ng sha1:4bd63e1e24faa047649891d9db6ebce3ba8ff988 /etc/sudoers.d/README
11 cd0f01b02a6fddea4275d422d1bbfc5596a684bd ima-ng sha1:8913365eeb76f1061c777b554748807c51838ceb /etc/passwd
11 4d3bb1a1ec24e2ae7784fb8b1c1f1bce9358a252 ima-ng sha1:7be0c4f402d2c0e120e1be86ffb86c8490b091a4 /etc/host.conf
11 8baaff611b8a9c6a14518876236079fef643b491 ima-ng sha1:04ebc8db5f1d558f6c1de3b6063412a995c435bd /run/resolvconf/resolv.conf
11 fd83e9c451e12b72920fed9f4b9cfe9bc6a54e5c ima-ng sha1:14bdc86bdf42ca611e69de9df0d4e28af88d3367 /etc/hosts
```

Included in 4.8

3. Preserving the IMA Log

Problem: How can measurements from a previous kernel be validated?

Solution:

1. Serialize the IMA log
2. Store in the kexec image
3. (reboot)
4. IMA restores log from physical memory

3. IMA Log (cont.)

Implementation for POWER

1. `kexec_file_load`:
   - Allocate buffer (kexec segment) for log serialization
   - Create device tree entry for the start and ending address
   - Set reboot hook
2. Reboot hook:
   - Serialize the log
   - Store into kexec image
3. Immediately Before Purgatory:
   - Move log from kexec image to physical memory address
4. New kernel boot:
   - IMA checks for device tree entry
   - If exists, restores log

4. Device Tree

Problem: Need to measure and pass the device tree through `kexec_file_load`

Solution: ¯_(ツ)_/¯
Current questions:

- What should be measured? (or can be?)
- What should the userspace be allowed to supply?

Questions?